**Network Security Lab 3**

**Buffer Overflow**

0656511黃誠發

1. 目標：找到藏在server上面的flag 1和flag 2
2. 第一步是先進到server看裏面有什麼東西，會發現裏面有三種模式可以選擇：
   * 第一個是查看所有人的資料
   * 第二個是修改所有人的資料
   * 第三個是離開
3. 再去看提供的source code裡面的main function會call func()
   * ret\_addr會將func() 的return address記錄下來，而這是我們攻擊的第一關鍵
   * 發現在func() while loop的case 3會做return address的比對來決定要return到哪個function
4. 在func() 中的variable:
   * ret\_addr透過 proj3.asm 會存在 ebp-0x88 的位址上
   * 會往上疊variable secret.info 透過 struct 得知 32bytes 因爲這是在 32位元作業系統，所以 char array 開 32 格子所以是 32 bytes 換成 16 進位就會變成 0x20 剛好就是 proj3.asm 的 ebp-0x64。
5. 接下來會有 4 bytes 的 secret，這是一個每次進來都不一樣的 random number，回到剛剛的 while loop 發現我們可以透過 case 2 需要 secret 才能進入修改內容，再來可以看到 edit\_note 這個 function 一開始需要瞭解 STU 這個 struct，他是由 16 bytes name, 16 bytes note, 8 bytes stuid, 4 bytes age 共 44 bytes 從 func() 的 proj3.asm 就可以找到 S是從 ebp-0x60 開始往上疊 88 bytes(S有 2 個人資料)，下圖爲 stack 大致架構圖，只知道這些後就可以開始想辦法攻擊
6. 4. 首先，要做 buffer overflow 攻擊，我們必須找到可以寫入字串的地方才可以塞一堆垃圾訊息使buffer overflow 讓return address導到錯的地方（我們想要的地方 magic1 的位址）
7. 5. menu 的第二個部分就是唯一可以寫入的地方，而透過剛剛的 stack 示意圖和 source code ，可以發現要進入 edit 需要 secret 但是在 view function 中有一個致命的漏洞，在 C 語言中若宣告爲 int 則該變數可爲正數或負數，由於剛好secret 存放的位址剛好在 S[0] 下面，所以我們只要先進入 view info 後 id 輸入 -1 則會顯示下面的狀態，如此就能得到 secret，而爲什麼會出現在 age 的欄位是因爲secret 剛好是 ebp-0x60 往下讀 44 bytes 的最上面的 4bytes 緊接着 S[0]。
8. 6. 接着我進入 edit info 輸入剛剛取得的 secret 進入第二關，一樣選擇任意使用者，進入第三層就是要來輸入 overflow 字串的地方，我發現只輸入小於16 的正整數根本沒辦法找到真正 return address 的位址，由於現在進入 struc stu\_info note 部分然而有 16 bytes name 在下面所以我們可以從S[0] 的位址往上推得note 位址爲 ebp-0x50，這樣只要往上塞 5\*16+4（存 ebp 的大小） bytes 就可以找到 return address 的位址。
9. 7. 最後去找 proj3.asm 的 magic1 在哪裏覆蓋掉原本的 address 再回到選擇 3 就可以順利取得 flag1
10. 8. 因爲我不知道怎麼直接用 echo 寫檔的方式把取得 secret 存下來再輸入進入編輯模式，所以我寫成腳本讓他跑。